Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Please amend claims 14, 18, 19, and 21-25 as follows:

Claims 1-13 (canceled).

14. (currently amended): An array processor comprising:

a physical ~~configuration~~ MxN array organization of at least two processing elements; and

a processor state register storing a context status bit (CSB), the CSB having a first state and a second state, each processing element operating to detect the state of the CSB,

the array processor upon detection of the first state of the CSB operating in a first operating context adapted for processing a first software task where the first software task is written for an MxN operating configuration which matches the physical ~~configuration~~ MxN array organization, where M represents the number of rows of processing elements and N represents the number of columns of processing elements,

the array processor upon detection of the second state of the CSB operating in a second operating context adapted for a second software task where the second software task is written for a second array processor having an OxP operating configuration where O is the number of rows of processing elements and P is the number of columns of processing elements, the OxP operating configuration not matching ~~different~~ the physical ~~configuration~~MxN array organization as either M≠O, N≠P, or M≠O and N≠P.

15. (previously presented): The array processor of claim 14 wherein in the first operating context, the array processor utilizes a first and a second set of register files, the first set of register files associated with one of the processing elements and the second set of register files associated with another of the processing elements.

16. (previously presented): The array processor of claim 15 further comprising:

an eventpoint mechanism to trigger storing the data contents of the first set of register files in the background while the first software task uses the second set of register files in the foreground.

17. (previously presented): The array processor of claim 15 further comprising:

an eventpoint mechanism to trigger loading the first set of register files in the background while the first software task uses the second set of register files in the foreground.

18. (currently amended): The array processor of claim 14 wherein each processing element of the at least two processing elements has a physical identifier and a virtual identifier, wherein during the processing of the first software task, instructions are operable in each processing element according to its physical identifier, wherein during the processing of the second software task, instructions are ~~loaded to~~operable in each processing element according to its virtual identifier.

19. (currently amended): A method for providing reconfiguration of a first array processor having a ~~first~~ physical MxN array ~~configuration~~organization to emulate operation of a second array processor having a ~~second~~ physical OxP array ~~configuration~~organization where M and O represent the number of rows of processing elements and N and P represent the number of columns of processing elements, the method comprising:

providing the first array processor having at least two processing elements arranged in the physical MxN array organization;

storing a context status bit (CSB), the CSB having a first state and a second state;

detecting the state of the CSB;

upon detection of the first state, operating in a first operating context adapted for processing a first software task, wherein the first software task is written for the ~~first~~ physical MxN array ~~configuration~~organization; and

upon detection of the second state, operating in the second operating context adapted for processing a second software task, wherein ~~a~~the second software task is written for the ~~second~~ physical OxP array ~~configuration~~organization, where either M≠O or N≠P.

20. (previously presented): The method of claim 19 wherein the operating in the second operating context step further comprises:

setting the CSB to the first state; and

returning the processing to the first operating context.

21. (currently amended): The method of claim 19 wherein the ~~first~~ physical MxN array ~~configuration~~ organization comprises a 1x1 layout and the emulated ~~second~~ physical OxP array ~~configuration~~ organization comprises a 1x0 layout, wherein the 1x0 layout defines a sequence processor (SP) executing sequential instructions.

22. (currently amended): The method of claim 19 wherein the ~~first~~ physical MxN array ~~configuration~~organization comprises a 1x2 layout and the emulated ~~second~~ physical OxP array ~~configuration~~organization comprises a 1x1 layout.

23. (currently amended): The method of claim 19 wherein the ~~first~~ physical MxN array ~~configuration~~organization comprises a 1x5 layout and the emulated ~~second~~ physical OxP array ~~configuration~~organization comprises a 2x2 layout.

24. (currently amended): An apparatus for providing efficient sharing of programming resources in a merged very long instruction word (VLIW) sequence processor (SP) and VLIW processor element (PE) processor, the merged VLIW SP/PE processor operating to configur~~able~~ an array processor to operate in an ~~first merged processor~~ MxN operating configuration or in an ~~second merged processor~~ OxP operating configuration, where M and O are the number of rows of processing elements and N and P are the number of columns of processing elements, and where either M≠O or N≠P, the apparatus comprising:

an SP resource file having a first set of registers;

a PE resource file having a second set of registers;

an input for receiving a VLIW presented for execution, the VLIW having at least two instructions, each instruction encoded with a different setting of an SP/PE-bit; and

a processor state register storing a context select bit (CSB), the merged VLIW SP/PE processor reading the values of the CSB and the SP/PE-bit of an instruction, the value of the CSB ~~to~~ selecting ~~a~~ the ~~first merged processor~~MxN operating configuration or ~~a~~the ~~second merged processor~~OxP operating configuration when processing the instruction, the ~~first merged processor~~MxN operating configuration adapted for accessing at least one register from the second set of registers when processing an SP instruction, the value of the S/P bit determining whether an instruction is executed in the merged VLIW SP/PE processor or is executed in an array of processing elements defined by the selected operating configuration.

25. (currently amended): The apparatus of claim 24 wherein the ~~second merged processor~~OxP operating configuration adapted for accessing at least one register from the first set of registers when processing an SP instruction and accessing at least one register from the second set of registers when processing a PE instruction.

26. (previously presented): The apparatus of claim 24 wherein the SP resource file is an SP register file, an SP address register file, or a SP machine state register file.

27. (previously presented): The apparatus of claim 24 wherein the PE resource file is a PE register file, PE address register file, or a PE machine state register files.

28. (previously presented): The apparatus of claim 24 further comprising:

at least two execution units associated with the at least two instructions in the VLIW; and

a plurality of multiplexers connected to the SP and PE resource files for selecting resource files from which the at least two execution units read data and to which the at least two execution units write data, a portion of the plurality of multiplexers associated with an execution unit controlled by a logical combination of the SP/PE bit and the CSB.

29. (previously presented): The apparatus of claim 24 wherein the VLIW SP processor and VLIW PE processor are indirect VLIW processors.